7-2013

On the effect and remedies of shrinkage on classification probability estimation

Zhengxiao WU
Singapore Management University, zxwu@smu.edu.sg

Yufeng LIU

Zhengxiao WU

DOI: https://doi.org/10.1080/00031305.2013.817356

On the Effect and Remedies of Shrinkage on Classification Probability Estimation

Chong ZHANG, Yufeng LIU, and Zhengxiao WU

Shrinkage methods have been shown to be effective for classification problems. As a form of regularization, shrinkage through penalization helps to avoid overfitting and produces accurate classifiers for prediction, especially when the dimension is relatively high. Despite the benefit of shrinkage on classification accuracy of resulting classifiers, in this article, we demonstrate that shrinkage creates biases on classification probability estimation. In many cases, this bias can be large and consequently yield poor class probability estimation when the sample size is small or moderate. We offer some theoretical insights into the effect of shrinkage and provide remedies for better class probability estimation. Using penalized logistic regression and proximal support vector machines as examples, we demonstrate that our proposed refit method gives similar classification accuracy and remarkable improvements on probability estimation on several simulated and real data examples.

KEY WORDS: Bias; High dimension; Refit; Regularization.

1. INTRODUCTION

Classification is an important tool for information extraction. It is an example of supervised learning techniques. The goal is to build a classification model, namely, a classifier, using the training data where both covariates and response labels are available. Once the classifier is obtained, it can be used for class prediction of new data points with only covariates observed. Classification can be applied in various fields ranging from artificial intelligence to econometrics, health study, and cancer research. Many classification techniques are available in the literature. See Hastie, Tibshirani, and Friedman (2009) for a comprehensive review of various classification methods. Commonly used classical methods include logistic regression, Fisher linear discriminant analysis, and nearest neighbors. These methods often work well in the traditional setting when the dimension is relatively low. However, recent technology has enabled us to gather data with very high dimensions. For example, DNA microarray technology measures tens of thousands of genes at the same time. High-dimensional and complex data pose challenges for the development of suitable statistical techniques. Recently, several classification techniques, originally introduced in the machine learning community, have become popular partially due to their ability to handle high-dimensional data. Important examples include Support Vector Machines (SVMs; Wahba 1999; Lin 2002) and Boosting (Freund and Schapire 1997).

Classification accuracy is one of the most important measures of classification performance. An accurate classifier can produce good prediction of class membership for new subjects. Another important issue is class probability estimation. As there are random errors involved in the class prediction, class probability estimation gives users information on how strong the evidence of classifying one subject into a particular class is. The problem of class probability estimation can be even more important than classification accuracy. For example, in disease diagnosis, it is vital for doctors and patients to know the chance of a certain disease instead of just a prediction. For two patients classified into the disease positive class, if their respective class probabilities are 0.51 and 0.99, the probability information is undoubtedly critical.

Before proceeding, we would like to clarify that traditionally, one may distinguish classification and regression in terms of how the data were obtained. In particular, in regression problems, it is common to have iid data from a joint distribution of covariates and response. For classification, one may obtain data for each class separately to ensure certain proportion of each class. In this article, our notion of classification is quite general as in Wahba (1999) and Hastie, Tibshirani, and Friedman (2009). When the response variable is categorical, we call the corresponding problem as a classification problem. In that sense, one may use some regression techniques to solve certain classification problems, such as using least-square regression to handle certain binary classification problems.

Our focus is on margin-based, sometimes called large margin, classification techniques. Such techniques can typically be written as a regularization problem of minimizing *Loss* + *Penalty*. Here, the loss term is used to ensure goodness of fit of the resulting model on the training data. The penalty term, also known as the regularization term, prevents overfitting through shrinkage so that the resulting model can produce accurate predictions. Many important classification techniques fit into the regularization framework, for example, Penalized Logistic

Chong Zhang (E-mail: *chongz@email.unc.edu*) is Ph.D. student and Yufeng Liu (E-mail: *yfliu@email.unc.edu*) is Professor, Department of Statistics and Operations Research, Carolina Center for Genome Sciences, Department of Biostatistics, University of North Carolina, Chapel Hill, NC 27599. Zhengxiao Wu is Assistant Professor, Department of Statistics and Applied Probability, National University of Singapore, Singapore (E-mail: *stawz@nus.edu.sg*). Zhang and Liu are partially supported by NSF Grant DMS-0747575. Wu is partially supported by MOE grant R155-000-105-112 and R155-000-126-112. The authors are indebted to two editors, the associate editor, and two referees, whose helpful comments and suggestions led to a much improved presentation.

Regression (PLR; Lin et al. 2000), AdaBoost in Boosting (Freund and Schapire 1997; Friedman, Hastie, and Tibshirani 2000), Import Vector Machine (Zhu and Hastie 2005), Proximal SVM (PSVM; Suykens and Vandewalle 1999; Fung and Mangasarian 2001; Tang and Zhang 2005), ψ-learning (Shen et al. 2003), and more recently, Large Margin Unified Machines (Liu, Zhang, and Wu 2011). The regularization term is especially important for high-dimensional data analysis.

One can use the loss function, or sometimes the related likelihood function, to derive a formula for probability estimation using the classification function. For example, in PLR, one can use the inverse logit transformation to estimate the probability. Lin (2000) showed that under certain conditions, the PLR probability estimator converges to the true probability as the training sample size gets large. It is common in practice to use such an estimator of probability, without taking the shrinkage effect into account. In this article, we demonstrate that when the sample size n is relatively small compared to the dimension d, the shrinkage effect can be very large on probability estimation. In practice, such probability estimators can have sizeable biases and consequently may give misleading results. Our goal is to explore the shrinkage effect on classification and more importantly on probability estimation. Through theoretical studies, we demonstrate how shrinkage affects probability estimation in binary classification problems. In particular, we show that shrinkage tends to force the resulting naive probability estimator toward $1/2$ in standard learning, where both classes are treated equally. Inspired by this phenomenon, we explore new methods to achieve better probability estimation. Our proposed refit method is shown to give consistent probability estimation, and works remarkably well in the numerical examples. For illustration, we focus on PLR and PSVM in this article, however, our idea is applicable to general margin-based classifiers as well.

In Section 2, we review large margin classification techniques and explore some theoretical properties of shrinkage for several methods. Our results shed some light on poor probability estimation without adjusting the shrinkage effect. Both standard and weighted learning settings are considered. In Section 3, we propose a new two-stage refit method for better probability estimation. Given the classification function of the penalized method from the first step, we refit the classifier as a one-dimensional problem without penalization to correct the shrinkage bias. We show that the refit method often has a large gain in probability estimation, while keeping similar classification performance to the first step. Some asymptotic consistency results of the refit procedure are provided as well. In Section 4, we use simulated examples to examine the performance of the refit approach. In Section 5, we evaluate the methods on a real data example. Some discussion is provided in Section 6. All technical proofs are collected in the Appendix.

2. LARGE MARGIN CLASSIFIERS AND THE SHRINKAGE EFFECT

2.1 Framework of Large Margin Classifiers

In supervised learning, we have a training dataset $\{(\boldsymbol{x}_i, y_i); i = 1, \ldots, n\}$ that contains n observations, where $\boldsymbol{x}_i \in R^d$ is a d-dimensional covariate vector and y_i is the response variable. When y is a continuous variable, we have the well-known regression problem. In that case, it is common to assume that the data are iid observations according to an unknown probability distribution $P(\boldsymbol{x}, y)$ and the goal is to estimate $E(y|\boldsymbol{x})$. When y is categorical, we then have a classification problem using our broad definition of classification in Section 1. Our focus in this article is on binary classification with $y \in \{\pm 1\}$. If the data are iid as in the typical regression setting, one can use a regression technique to estimate $E(y|\boldsymbol{x}) = 2P(y = 1|\boldsymbol{x}) - 1$ directly.

For classification problems, it is common to have independent samples for each class, obtained from $P(\boldsymbol{x}|y)$. The sample class proportions, however, can be different from population class proportions. Assume that π and $1 - \pi$ are the proportions of positive and negative classes in the population, respectively. Similarly, π_s and $1 - \pi_s$ are the class proportions of the sample. Then the joint distribution for the sample is $P_s(\boldsymbol{x}, y) = P(\boldsymbol{x}|y = 1)\pi_s + P(\boldsymbol{x}|y = -1)(1 - \pi_s)$ and the population joint distribution is $P(\boldsymbol{x}, y) = P(\boldsymbol{x}|y = 1)\pi + P(\boldsymbol{x}|y = -1)(1 - \pi)$. When there is sampling bias with $\pi \neq \pi_s$, one needs to make adjustments after obtaining estimation for $P_s(y = 1|\boldsymbol{x})$. In particular, we can show that the population odds $P(y = 1|\boldsymbol{x})/(1 - P(y = 1|\boldsymbol{x}))$ and the sample odds $P_s(y = 1|\boldsymbol{x})/(1 - P_s(y = 1|\boldsymbol{x}))$ satisfy that

$$\frac{P(y = 1|\boldsymbol{x})}{1 - P(y = 1|\boldsymbol{x})} = \frac{P_s(y = 1|\boldsymbol{x})}{1 - P_s(y = 1|\boldsymbol{x})} \frac{(1 - \pi_s)\pi}{\pi_s(1 - \pi)}.$$

For simplicity, we first assume both the sample and population are from the same distribution $P(\boldsymbol{x}, y)$. When there is sampling bias, we can use weighted learning in Section 2.3 to adjust the sampling bias.

To classify a new input vector $\boldsymbol{x}$, a classification (discrimination) function f is estimated from the training dataset, and $\text{sign}[f(\boldsymbol{x})]$ is used as the predicted label. Because of the sign function used as the classification rule, the quantity $yf(\boldsymbol{x})$, called the functional margin, is very important. In particular, $yf(\boldsymbol{x}) > 0$ indicates correct classification of the point $(\boldsymbol{x}, y)$. A margin-based classifier uses the functional margin $yf(\boldsymbol{x})$ in its corresponding regularization problem. Due to the goal of having large margin $yf(\boldsymbol{x})$, it is also known as a large margin classifier (Cristianini and Shawe-Taylor 2000; Hastie, Tibshirani, and Friedman 2009).

In the regularization framework, we solve the following optimization problem:

$$\min_{f \in \mathcal{F}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell[y_i f(\boldsymbol{x}_i)] + \lambda J(f) \right\}, \tag{1}$$

where $\ell(\cdot)$ is a loss function that uses the functional margin to ensure goodness of fit of the model on the training data, $\mathcal{F}$ is the functional space of interest, and $J(f)$ is a regularization term on f to avoid overfitting. For example, in linear learning, $\mathcal{F}$ consists of the set of linear functions. The tuning parameter λ balances the two terms in (1) to ensure good generalization abilities of the resulting classifier for future prediction. A proper choice of λ is very important.

The loss function $\ell(\cdot)$ is typically prespecified and differs among various methods. For example, the deviance loss $\ell(u) = \log(1 + e^{-u})$ leads to the PLR, AdaBoost is shown to be approximately equivalent to using the exponential loss $\ell(u) = e^{-u}$ (Friedman, Hastie, and Tibshirani 2000), the hinge loss yields the SVM $\ell(u) = (1-u)_+$, and the squared error loss $\ell(u) = (1-u)^2$ gives the PSVM (Fung and Mangasarian 2001), which is essentially equivalent to performing penalized least-square linear regression of the binary response $Y \in \{\pm 1\}$ on $\boldsymbol{x}$ in linear learning. The squared error loss has a close connection with linear discriminant analysis. In particular, if the class label Y is coded using $\{0,1\}$, then minimizing the empirical squared errors without regularization leads to Fisher's linear discriminant function (see Williams 1959 and chap. 4 of Hastie, Tibshirani, and Friedman 2009).

As different loss functions yield various methods, it is essential to study the properties of these loss functions. Many loss functions are Fisher consistent (Lin 2004; Bartlett, Jordan, and McAuliffe 2006). For a standard binary classification problem, the corresponding loss function $\ell(\cdot)$ is Fisher consistent if and only if $\operatorname{sign}[f^*(\boldsymbol{x})] = \operatorname{sign}[p(\boldsymbol{x}) - \frac{1}{2}]$, where $f^*(\boldsymbol{x}) = \operatorname{argmin}_f E\{\ell[Yf(\boldsymbol{X})]|\boldsymbol{X} = \boldsymbol{x}\}$ and $p(\boldsymbol{x}) = P(Y = +1|\boldsymbol{X} = \boldsymbol{x})$. Thus, Fisher consistency essentially ensures that the population minimizer of the loss function have the same sign function as $p(\boldsymbol{x}) - 1/2$. In practice, as sample size increases, the resulting classification boundary approaches the theoretically optimal boundary, that is, the Bayes' decision boundary $\{\boldsymbol{x} : p(\boldsymbol{x}) = 1/2\}$. Note that the terminology of Bayes' decision boundary is a commonly used term to refer to the best theoretical classification boundary in the literature, and the corresponding smallest error is known as the Bayes' error (Hastie, Tibshirani, and Friedman 2009). Fisher consistency is a weak requirement on the loss function of a classifier. Lin (2004) showed that a loss function $\ell(\cdot)$ is Fisher consistent if it satisfies

A.1. $\ell(u) < \ell(-u), \forall u > 0$.
A.2. $\ell'(0)$ exists, where $\ell'(u)$ is the derivative of $\ell(u)$.

All the aforementioned loss functions satisfy A.1 and A.2 and thus are all Fisher consistent.

Ideally, we would like to transform the classification function $f(\boldsymbol{x})$ to estimate the class conditional probability $p(\boldsymbol{x})$. Thus, once $\hat{f}(\boldsymbol{x})$ is obtained, we estimate $p(\boldsymbol{x})$ accordingly. Our goal is to explore a general Fisher consistent loss function $\ell(\cdot)$ and investigate conditions for us to estimate $p(\boldsymbol{x})$ through some transformation of $f(\boldsymbol{x})$.

To estimate $p(\boldsymbol{x})$ from $f(\boldsymbol{x})$, a natural condition on $\ell(\cdot)$ is to have a one-to-one mapping between $f^*(\boldsymbol{x})$ and $p(\boldsymbol{x})$. Theorem 1 provides conditions on $\ell(\cdot)$ so that such a one-to-one correspondence exists. Note that a similar theorem with different assumptions was developed in Zou, Zhu, and Hastie (2008).

Theorem 1. The following conditions are sufficient for the minimizer $f^*(\boldsymbol{x}) = \operatorname{argmin}_f E\{\ell[Yf(\boldsymbol{X})]|\boldsymbol{X} = \boldsymbol{x}\}$ and $p(\boldsymbol{x})$ to have a one-to-one correspondence:

B.1. $\ell(u)$ is twice differentiable with respect to u.
B.2. $\ell'(u) + \ell'(-u) < 0, \forall u$.
B.3. $\ell'(u)\ell''(-u) + \ell'(-u)\ell''(u) < 0$ for any u, where $\ell''(u)$ is the second derivative of $\ell(u)$.

Under these conditions, the mapping between $f^*(\boldsymbol{x})$ and $p(\boldsymbol{x})$ is $p(\boldsymbol{x}) = \frac{\ell'[-f^*(\boldsymbol{x})]}{\ell'[f^*(\boldsymbol{x})] + \ell'[-f^*(\boldsymbol{x})]}$.

Most of the loss functions mentioned earlier, for example, the deviance loss, the exponential loss, and the squared error loss, satisfy the conditions in Theorem 1. Consequently, the corresponding class probability $p(\boldsymbol{x})$ can be estimated using the relationship between $p(\boldsymbol{x})$ and $f^*(\boldsymbol{x})$ provided in Theorem 1. For PLR, $f^*(\boldsymbol{x}) = \operatorname{logit}(p(\boldsymbol{x}))$ and we then use the inverse logit transformation on $f(\boldsymbol{x})$ to estimate $p(\boldsymbol{x})$. For the hinge loss of SVM, $f^*(\boldsymbol{x}) = \operatorname{sign}[p(\boldsymbol{x}) - 0.5]$, and one cannot estimate $p(\boldsymbol{x})$ directly.

When the conditions of Theorem 1 hold, we call the solution

$$p_0(f) = \frac{\ell'(-f)}{\ell'(f) + \ell'(-f)}, \tag{2}$$

the original method. Once $\hat{f}(\boldsymbol{x})$ is obtained, the estimator of $p(\boldsymbol{x})$ using the original method becomes $p_0[\hat{f}(\boldsymbol{x})]$. For some loss functions such as the squared error loss, $p_0[\hat{f}(\boldsymbol{x})]$ may be outside of [0, 1]. When $p_0[\hat{f}(\boldsymbol{x})] < 0$, or > 1, it is typically set to be 0 or 1, respectively. Alternatively, one may consider a restricted functional space $\mathcal{F}$ to ensure the resulting $p_0[\hat{f}(\boldsymbol{x})] \in [0, 1]$. For example, $\hat{f}(\boldsymbol{x})$ estimates $E(y|\boldsymbol{x}) = 2p(\boldsymbol{x}) - 1$ for the case of squared error loss, and one can restrict $f(\boldsymbol{x}) \in [0, 1]$ for any $f \in \mathcal{F}$ to ensure appropriate estimation of $p(\boldsymbol{x})$. However, such a restriction can lead to a nonconvex minimization problem and make the corresponding implementation challenging.

Notice that the approach using (2) for probability estimation pays attention only to the first term in (1) for class probability estimation. Asymptotically the original probability estimator is consistent under various conditions (Lin 2000). However in practice, when the sample size is moderate or small, the shrinkage effect from the regularization term in (1) can be large. Consequently, the original method for probability estimation can be severely biased. We will demonstrate the shrinkage effect both theoretically and numerically. In Sections 2.2 and 2.3, we will explore the theoretical impact of shrinkage on class probability estimation for both standard and weighted learning settings.

2.2 Theoretical Property of Shrinkage

As we pointed out earlier, ignoring the effect of the regularization term $J(f)$ in (1) may create bias in class conditional probability estimation. Next we explore how this regularization term creates shrinkage on the estimation of the classification function $f(\boldsymbol{x})$, which leads to a large gap between the true class conditional probability and its original estimation. For simplicity, we consider linear learning with $f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\beta}$. When the linear functional space spanned by $\boldsymbol{x}$ is insufficient, one may consider a higher, possibly infinite, dimensional space spanned by $\phi(\boldsymbol{x})$, where $\phi(\cdot)$ is the mapping of $\boldsymbol{x}$ from the linear space to a higher dimensional space. One may specify $\phi(\boldsymbol{x})$ explicitly, and perform learning with $f(\boldsymbol{x}) = \phi(\boldsymbol{x})^T\boldsymbol{\beta}$. Such an approach can be difficult to implement when an infinite-dimensional space is needed. Another approach is to perform the mapping implicitly using the so-called *kernel* trick, with

$K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \langle \phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2) \rangle$, where $K(\cdot, \cdot)$ is a *kernel* function. With a given kernel function, one can perform kernel learning without explicitly specifying $\phi(\cdot)$. More details about the kernel learning and kernel trick can be found in Cristianini and Shawe-Taylor (2000), Schölkopf and Smola (2002), and Wahba (1999). The Gaussian kernel is a commonly used nonlinear kernel with $K(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp(-\frac{(\boldsymbol{x}_1 - \boldsymbol{x}_2)^T(\boldsymbol{x}_1 - \boldsymbol{x}_2)}{\sigma^2})$, where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are two covariates in the original space and σ is a fixed constant. Our idea and method can be directly extended to the kernel framework, and we do not include the details here.

In the linear learning setup, we assume that the first coordinate of $\boldsymbol{x}$ corresponds to the constant term and as a result, the first element of $\boldsymbol{\beta}$ represents the intercept term β_0 of the linear function. In our theoretical exploration, for simplicity, we let $J(f) = \|\boldsymbol{\beta}\|^2 = \boldsymbol{\beta}^T\boldsymbol{\beta}$ be the regularization term. Note that here $J(f)$ includes β_0 as well. In practice the intercept is often not penalized.

To explore the effect of shrinkage, ideally, we should study $\operatorname{argmin}_f E\{\ell[Yf(\boldsymbol{X})] + J(f)\}$. However, we cannot derive the solution directly since it depends on the underlying distribution $P(\boldsymbol{x}, y)$. Instead, we consider the conditional minimizer

$$f^{**}(\boldsymbol{x}) = \operatorname*{argmin}_f E\{\ell[Yf(\boldsymbol{X})] + J(f) | \boldsymbol{X} = \boldsymbol{x}\}.$$

Notice that this definition of $f^{**}(\boldsymbol{x})$ is pointwise in terms of $\boldsymbol{x}$, just as $f^*(\boldsymbol{x})$. In linear learning with $f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\beta}$ and $J(f) = \|\boldsymbol{\beta}\|^2 = \boldsymbol{\beta}^T\boldsymbol{\beta}$, this is equivalent to finding

$$\boldsymbol{\beta}^{**}(\boldsymbol{x}) = \operatorname*{argmin}_{\boldsymbol{\beta}} E\{\ell[Y\boldsymbol{X}^T\boldsymbol{\beta}] + \lambda\|\boldsymbol{\beta}\|^2 | \boldsymbol{X} = \boldsymbol{x}\}. \tag{3}$$

Note that the solution $\boldsymbol{\beta}^{**}(\boldsymbol{x})$ in (3) depends on $\boldsymbol{x}$ so $\boldsymbol{x}^T\boldsymbol{\beta}^{**}(\boldsymbol{x})$ is not a linear classifier. Although our derivation is conditioned on $\boldsymbol{x}$, it can help to reveal the effect of shrinkage on probability estimation.

To calculate (3), define $S[\boldsymbol{\beta}(\boldsymbol{x})] = E[\ell(Y\boldsymbol{X}^T\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|^2 | \boldsymbol{X} = \boldsymbol{x}] = p(\boldsymbol{x})\ell[\boldsymbol{x}^T\boldsymbol{\beta}(\boldsymbol{x})] + [1 - p(\boldsymbol{x})]\ell[-\boldsymbol{x}^T\boldsymbol{\beta}(\boldsymbol{x})] + \lambda\|\boldsymbol{\beta}(\boldsymbol{x})\|^2$. To minimize $S[\boldsymbol{\beta}(\boldsymbol{x})]$, we solve $\frac{\partial S[\boldsymbol{\beta}(\boldsymbol{x})]}{\partial \boldsymbol{\beta}(\boldsymbol{x})}|_{\boldsymbol{\beta}(\boldsymbol{x}) = \boldsymbol{\beta}^{**}(\boldsymbol{x})} = 0$. Thus, we have

$$p(\boldsymbol{x})\ell'[\boldsymbol{x}^T\boldsymbol{\beta}^{**}(\boldsymbol{x})]\boldsymbol{x} - [1 - p(\boldsymbol{x})]\ell'[-\boldsymbol{x}^T\boldsymbol{\beta}^{**}(\boldsymbol{x})]\boldsymbol{x} + 2\lambda\boldsymbol{\beta}^{**}(\boldsymbol{x}) = 0,$$

which is equivalent to

$$p(\boldsymbol{x})\boldsymbol{x} = \frac{\ell'[-f^{**}(\boldsymbol{x})]}{\ell'[-f^{**}(\boldsymbol{x})] + \ell'[f^{**}(\boldsymbol{x})]}\boldsymbol{x} - \frac{2\lambda}{\ell'[-f^{**}(\boldsymbol{x})] + \ell'[f^{**}(\boldsymbol{x})]}\boldsymbol{\beta}^{**}(\boldsymbol{x}). \tag{4}$$

Let $A[f^{**}(\boldsymbol{x})] = -\frac{2}{\ell'[-f^{**}(\boldsymbol{x})] + \ell'[f^{**}(\boldsymbol{x})]}$. Using the definition of $A[f^{**}(\boldsymbol{x})]$ and (2), (4) becomes

$$p(\boldsymbol{x})\boldsymbol{x} = p_0[f^{**}(\boldsymbol{x})]\boldsymbol{x} + \lambda A[f^{**}(\boldsymbol{x})]\boldsymbol{\beta}^{**}(\boldsymbol{x}). \tag{5}$$

Note that both $p_0[f^{**}(\boldsymbol{x})]$ and $A[f^{**}(\boldsymbol{x})]$ are scalars. Since $p(\boldsymbol{x})$ is fixed for a given $\boldsymbol{x}$, to have (5) hold, $\boldsymbol{\beta}^{**}(\boldsymbol{x})$ satisfies $\boldsymbol{\beta}^{**}(\boldsymbol{x}) = c(\boldsymbol{x})\boldsymbol{x}$, where $c(\boldsymbol{x}) = \frac{p(\boldsymbol{x}) - p_0[f^{**}(\boldsymbol{x})]}{\lambda A[f^{**}(\boldsymbol{x})]}$ is a scalar that depends on $\boldsymbol{x}$. This implies that $\boldsymbol{\beta}^{**}(\boldsymbol{x})$ is a function of $\boldsymbol{x}$ and it varies according to different $\boldsymbol{x}$ because we derive such a relationship for a fixed $\boldsymbol{x}$. However, in practice, we calculate a common $\boldsymbol{\beta}$ for all $\boldsymbol{x}$'s. Nevertheless, our derivation on each fixed $\boldsymbol{x}$ using conditional expectation helps to shed some light on the effect of shrinkage.

To further simplify (5), for any d-dimensional vector $\boldsymbol{z}$ with $\boldsymbol{z}^T\boldsymbol{x} \neq 0$, we have

$$p(\boldsymbol{x})\boldsymbol{z}^T\boldsymbol{x} = p_0[f^{**}(\boldsymbol{x})]\boldsymbol{z}^T\boldsymbol{x} + \lambda A[f^{**}(\boldsymbol{x})]\boldsymbol{z}^T\boldsymbol{\beta}^{**}(\boldsymbol{x}),$$

and consequently we obtain the expression of $p(\boldsymbol{x})$ as

$$p(\boldsymbol{x}) = p_0[f^{**}(\boldsymbol{x})] + \lambda A[f^{**}(\boldsymbol{x})]\frac{\boldsymbol{z}^T\boldsymbol{\beta}^{**}(\boldsymbol{x})}{\boldsymbol{z}^T\boldsymbol{x}}. \tag{6}$$

If we set $\boldsymbol{z} = \boldsymbol{x}$, with $\boldsymbol{\beta}^{**}(\boldsymbol{x}) = c(\boldsymbol{x})\boldsymbol{x}$, we have $c(\boldsymbol{x}) = \frac{\boldsymbol{\beta}^{**}(\boldsymbol{x})^T\boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}} = \frac{f^{**}(\boldsymbol{x})}{\|\boldsymbol{x}\|^2}$. Thus, $\operatorname{sign}[c(\boldsymbol{x})] = \operatorname{sign}[f^{**}(\boldsymbol{x})] = \operatorname{sign}\{p_0[f^{**}(\boldsymbol{x})] - 0.5\}$, where the last equality follows the fact that the function $p_0(\cdot)$ in (2) is strictly increasing and $p_0(0) = 0.5$. Thus, Equation (6) can be expressed as

$$p(\boldsymbol{x}) = p_0[f^{**}(\boldsymbol{x})] + \lambda A[f^{**}(\boldsymbol{x})] \cdot |c(\boldsymbol{x})| \cdot \operatorname{sign}\{p_0[f^{**}(\boldsymbol{x})] - 0.5\}, \tag{7}$$

where $A[f^{**}(\boldsymbol{x})] = -\frac{2}{\ell'[-f^{**}(\boldsymbol{x})] + \ell'[f^{**}(\boldsymbol{x})]} > 0$. Comparing to the formula of $p(\boldsymbol{x}) = p_0[f^{**}(\boldsymbol{x})]$ in Theorem 1, we have an extra term $t(\lambda) = \lambda A[f^{**}(\boldsymbol{x})] \cdot |c(\boldsymbol{x})| \cdot \operatorname{sign}\{p_0[f^{**}(\boldsymbol{x})] - 0.5\}$, which comes from the regularization term $J(f)$. Interestingly, $t(\lambda)$ has the same sign as $p_0[f^{**}(\boldsymbol{x})] - 0.5$. When $p_0[f^{**}(\boldsymbol{x})] > 0.5$, $p(\boldsymbol{x}) = p_0[f^{**}(\boldsymbol{x})] + t(\lambda) > p_0[f^{**}(\boldsymbol{x})]$. This implies that using $p_0[f^{**}(\boldsymbol{x})]$ underestimates $p(\boldsymbol{x})$. Similarly, $p_0[f^{**}(\boldsymbol{x})]$ overestimates $p(\boldsymbol{x})$ when $p_0[f^{**}(\boldsymbol{x})] < 0.5$. As a result, we can conclude that shrinkage will push the original probability estimation toward 0.5 for binary classifiers. This finding matches our intuition. In particular, the penalization term in regularization shrinks coefficients toward 0 and thus shrinks the classification function toward 0. Consequently, it shrinks the class conditional probabilities toward 0.5. In Section 4, we confirm this finding via simulation and show the large biases of original probability estimation.

One important issue we would like to point out is that the formula (7) is derived using conditional expression for $\boldsymbol{X} = \boldsymbol{x}$. Thus strictly speaking, (7) is a correct way to estimate $p(\boldsymbol{x})$ if we have a solution of $\boldsymbol{\beta}^{**}(\boldsymbol{x})$ specific for each $\boldsymbol{x}$. This is certainly not feasible. For practical problems as given in (1), we need to solve for a common estimate of $\boldsymbol{\beta}$ using n observations. Thus, (6) is not an applicable formula for the estimation of $p(\boldsymbol{x})$. In Section 3.1, we will introduce a simple refit method that works remarkably well. Next, we discuss the shrinkage effect on weighted learning.

2.3 Extension to Weighted Learning

So far, our focus has been on standard learning and we treat two classes equally. In this section, we study the extension of shrinkage effect to weighted learning. Weighted learning can be useful in many situations. Here, we briefly describe three scenarios: unequal costs, biased sampling, and unbalanced classification. Lin, Lee, and Wahba (2002) previously discussed nonstandard situations such as unequal costs and biased sampling for the SVM.

Unequal costs are needed for many practical problems. For example, wrongly classifying a patient with a fatal disease to the healthy group may be viewed as substantially more costly

than claiming the presence of the disease while it is not. In that case, unequal costs should be used to reflect the differences of these two types of misclassification.

Another important use of weighted learning is to adjust biased sampling. In many practical classification problems, the class proportions in the sample may be very different from those in the target population due to sampling bias. For example, if the two classes have very different proportions in the population, the smaller class may be oversampled, while the larger class may be undersampled so that the resulting sample can be more balanced. However, since we build the classifier based on the sample and predict classes of data from the population, this sampling bias can create problems. Weighted learning can be used to adjust such discrepancy.

Unbalanced classification is another case that weighted learning can be very effective. In standard learning it is common to evaluate the performance of a classifier by its overall prediction error rate. In real data applications, unbalanced classification problems can be challenging even when there is no sampling bias. For instance, in classifying patients into cancer versus noncancer groups, we could have 99% healthy patients and 1% cancer patients in the sample. In that case, we may have a naive classifier that predicts all patients into the healthy group with a 99% overall classification accuracy. To overcome this difficulty, one can use various weighted learning procedures (Qiao and Liu 2009).

Denote by $w(+)$ and $w(-)$, the weights for positive and negative classes, respectively. Then instead of (1), we solve the following optimization problem:

$$\min_{f\in\mathcal{F}} \left\{ \frac{1}{n}\sum_{i=1}^{n} W(y_i)\ell[y_i f(\boldsymbol{x}_i)] + \lambda J(f) \right\},$$

where $W(y_i) = w(+)$ if $y_i = +1$ and $W(y_i) = w(-)$ otherwise. Here $w(+)$ and $w(-)$ represent the weights for these two classes.

Denote $c(+1|-1)$ for the false-positive cost for points in the class -1 misclassified into the $+1$ class and similarly $c(-1|+1)$ for the false-negative cost for points in the class $+1$ misclassified into the -1 class. The costs of correct classification, that is, $c(+1|+1)$ and $c(-1|-1)$, are set to be 0. Then if the overall misclassification cost is used as the classification criterion, the optimal choice of $w(+)$ and $w(-)$ is that $w(+) = \frac{c(-1|+1)\pi}{\pi_s}$ and $w(-) = \frac{c(+1|-1)(1-\pi)}{1-\pi_s}$ (Qiao et al. 2010). Note that both costs and class proportions are used in the construction of weights. Furthermore, estimators can be used if the true proportions are not available. More details on the justification of these weights as well as different classification criteria can be found in Qiao et al. (2010).

Next we show that our developments in Section 2.2 can be directly extended to weighted learning. The following proposition illustrates the Bayes' boundary for weighted learning.

Proposition 1. (Wang, Shen, and Liu 2008) Assume A.1 and A.2 hold. Then the minimizer $f^*(\boldsymbol{x}) = \operatorname{argmin}_f E\{W(Y)\ell[Yf(\boldsymbol{X})]|\boldsymbol{X} = \boldsymbol{x}\}$ satisfies $\operatorname{sign}[f^*(\boldsymbol{x})] = \operatorname{sign}[p(\boldsymbol{x}) - \frac{w(-)}{w(+)+w(-)}]$.

From Proposition 1, we can see that the new Bayes' boundary for the population of interest incorporating the costs becomes $\{\boldsymbol{x} : p(\boldsymbol{x}) = \frac{w(-)}{w(+)+w(-)}\}$ for weighted learning. In Section 2.2, we show that with equal weights, the regularization term shrinks the probability estimation toward 1/2. In the weighted learning case, (4) becomes

$$p(\boldsymbol{x})\boldsymbol{x} = \frac{w(-)\ell'[-f^{**}(\boldsymbol{x})]}{w(-)\ell'[-f^{**}(\boldsymbol{x})] + w(+)\ell'[f^{**}(\boldsymbol{x})]}\boldsymbol{x} - \frac{2\lambda}{w(-)\ell'[-f^{**}(\boldsymbol{x})] + w(+)\ell'[f^{**}(\boldsymbol{x})]}\boldsymbol{\beta}^{**}(\boldsymbol{x}).$$

If we define $A[f^{**}(\boldsymbol{x})] = -\frac{2}{w(-)\ell'[-f^{**}(\boldsymbol{x})]+w(+)\ell'[f^{**}(\boldsymbol{x})]}$ accordingly, using similar derivations as in Section 2.2, one can verify that (7) becomes

$$p(\boldsymbol{x}) = p_0[f^{**}(\boldsymbol{x})] + \lambda A[f^{**}(\boldsymbol{x})] \cdot |c(\boldsymbol{x})| \cdot \operatorname{sign}\left\{ p_0[f^{**}(\boldsymbol{x})] - \frac{w(-)}{w(+)+w(-)} \right\}.$$

Thus, we can conclude that the regularization term now shrinks the probability estimation toward $\frac{w(-)}{w(+)+w(-)}$.

3. A NEW REFIT METHOD FOR PROBABILITY ESTIMATION

3.1 The Refit Procedure

Sections 2.2 and 2.3 demonstrate that although the shrinkage term in regularization helps to deliver accurate classification boundaries for large margin classifiers, it can adversely affect the accuracy of the probability estimation. Furthermore, it is difficult to derive an explicit correction term for the shrinkage effect on probability estimation. In this section, we propose a simple alternative to correct the biases introduced by shrinkage.

The idea of our refit method is as follows. In the linear case, we aim to estimate $\boldsymbol{\beta}$ so that $f = \boldsymbol{x}^T\boldsymbol{\beta}$ can yield class prediction based on whether $\operatorname{sign}(f) > 0$ or not. From Section 2.2, we learned that although shrinkage affects the size of f, it does not change the sign of f. This implies that we can get a good classification direction through $\hat{f}$, although the scale may be too small for probability estimation due to shrinkage. Our idea is to make use of the solution $\hat{f}$ from the large margin classifier and project the data on this direction. As long as the classification error is good, as it is typically the case for large margin classifiers, the corresponding projection direction should be reasonable as well. Based on these considerations, we propose to refit the data on the projected one-dimensional space without penalty.

We would like to point out that the idea of refit is not entirely new. It has been used in the regression setting to improve regression parameter estimation. In particular, Meinshausen (2007) suggested a two-step fitting algorithm, the relaxed Lasso, to alleviate the problem of bias in the regression parameter estimation. He proposed to first apply the regular Lasso method (Tibshirani 1996) to select a set of covariates as an "active set of variables," and then fit the Lasso again using the selected set of variables. The main idea of the relaxed Lasso is to eliminate some unimportant variables in the first step. Then, the amount of shrinkage needed in the second step will be much smaller, and consequently the resulting estimation bias can be alleviated compared to the original Lasso estimation. In contrast to regression, classification techniques aim to build accurate

classification boundaries. Once we get a good classification direction vector using the decision boundary, we can project the data on this one-dimensional space to correct bias using refitting. Unlike the relaxed Lasso that requires regularization on the second step as well, we refit the data without shrinkage.

As discussed earlier, a refit step without shrinkage has no risk of overfitting since the projected space is only one-dimensional. However, this refit step can correct the scale bias caused by shrinkage on the original fit. After the refit step, we then use the original method to estimate $p(\boldsymbol{x})$, based on the obtained $\hat{f}$ from the refit step.

Next we use the standard PLR to illustrate our refit method, although the idea is the same for many other methods as well.

Our proposed procedure for the refit PLR is summarized as follows:

Step 1. (Original fit) Fit PLR on the training data to obtain $\hat{f} = \boldsymbol{x}^T\hat{\boldsymbol{\beta}}$. Proper tuning on λ is needed.

Step 2. (Projection) Create a new training dataset on the projected space. The new training dataset contains $\{(\hat{\eta}_i, y_i); i = 1, \ldots, n\}$, where $\hat{\eta}_i = \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}$. Note that the new covariate space is only one-dimensional.

Step 3. (Refit) Fit the logistic regression without penalty on the new training data from Step 2 to get a new function $\hat{\hat{f}}(\hat{\eta}) = \hat{\gamma}_0 + \hat{\gamma}_1\hat{\eta}$.

Step 4. (Probability estimation) Our final probability estimation formula becomes $\hat{p}(\boldsymbol{x}) = \frac{e^{\hat{\hat{f}}(\boldsymbol{x})}}{e^{\hat{\hat{f}}(\boldsymbol{x})}+1}$, where $\hat{\hat{f}}(\boldsymbol{x}) = \hat{\gamma}_0 + \hat{\gamma}_1\boldsymbol{x}^T\hat{\boldsymbol{\beta}}$.

As we can see from the procedure, the refit method only adds small additional computational cost to the original PLR. The refit step is only a one-dimensional fit without penalty and can be done quickly. Furthermore, we suggest to refit the same model without regularization on the one-dimensional projection space.

The new parameters in the refit step serve as a correction on the scale bias created by shrinkage on the first step. As we will show in Section 3.2 and in simulation, the refit method gives almost identical classification errors as the original method, and at the same time it offers remarkable improvement on probability estimation.

For weighted learning, the refit steps are almost the same except some slight modifications needed for Steps 3 and 4. Once the projection in Step 2 is done, we refit the one-dimensional model with weighted learning using the original $w(+)$ and $w(-)$ as the weights, and obtain the corresponding probability estimator in Step 4. Using PLR as an example, the probability formula in Step 4 for weighted learning becomes $\hat{p}(\boldsymbol{x}) = \frac{w(-)e^{\hat{\hat{f}}(\boldsymbol{x})}}{w(-)e^{\hat{\hat{f}}(\boldsymbol{x})}+w(+)}$.

3.2 Theoretical Properties

In this section, we derive some asymptotic results for our refit method. In particular, we prove that under certain conditions, the refit procedure provides consistent probability estimation when the regularized method produces shrinkage on parameter estimation. For simplicity, we first focus on linear learning with equal weights and then discuss the results for weighted learning.

First we introduce some assumptions.

Assumption C.1. The loss function $\ell(\cdot)$ is convex and differentiable.

Assumption C.2. The distribution $P(\boldsymbol{x}, y)$ satisfies $P(Y = +1|\boldsymbol{X} = \boldsymbol{x}) = \frac{\ell'(-\boldsymbol{x}^T\boldsymbol{\beta}^*)}{\ell'(-\boldsymbol{x}^T\boldsymbol{\beta}^*)+\ell'(\boldsymbol{x}^T\boldsymbol{\beta}^*)} := p_0(\boldsymbol{x}^T\boldsymbol{\beta}^*)$, where $\boldsymbol{\beta}^*$ is the global minimizer of $E[\ell(Y\boldsymbol{X}^T\boldsymbol{\beta})]$ that does not depend on $\boldsymbol{X}$.

Assumption C.3. The estimated $\hat{\boldsymbol{\beta}} = \text{argmin}_{\boldsymbol{\beta}}[\frac{1}{n}\sum_{i=1}^n \ell(y_i\boldsymbol{x}_i^T\boldsymbol{\beta}) + \lambda J(\boldsymbol{\beta})]$ satisfies $\hat{\boldsymbol{\beta}} \to \theta\boldsymbol{\beta}^*$ in probability as $n \to \infty$, where $\theta \in (0, 1)$.

Next we discuss the use of these assumptions. For Assumption C.1, the convexity and differentiability are satisfied by many loss functions. Assumption C.2 ensures that the class conditional probability $P(Y = +1|\boldsymbol{X} = \boldsymbol{x})$ depends on $\boldsymbol{x}$ only through the inverse link function $\frac{\ell'(-\boldsymbol{x}^T\boldsymbol{\beta}^*)}{\ell'(-\boldsymbol{x}^T\boldsymbol{\beta}^*)+\ell'(\boldsymbol{x}^T\boldsymbol{\beta}^*)}$. For example, a similar assumption is used in logistic regression. Assumption C.3 deals with the asymptotic behavior of $\hat{\boldsymbol{\beta}}$. For many large margin classifiers, the direction of $\hat{\boldsymbol{\beta}}$ is usually close to that of $\boldsymbol{\beta}^*$, yielding a good classification boundary. However, as discussed in Section 2.2, the regularization term creates bias in the estimation of $\boldsymbol{\beta}^*$.

The next theorem justifies that the refit procedure helps to correct the scale bias introduced by the regularization term, while keeping the classification boundary almost the same.

Theorem 2. For linear learning, suppose that Assumptions A.1 and C.1–C.3 are satisfied. Then the estimates in Step 3 satisfy $\hat{\gamma}_0 \to 0$ and $\hat{\gamma}_1 \to \frac{1}{\theta}$ in probability, as $n \to \infty$.

From Theorem 2, we can see that the refit method asymptotically corrects the scale from shrinkage, thus provides consistent probability estimation.

For weighted learning, we have a similar result. In that case, we modify Assumptions C.2 and C.3 as follows, and an immediate corollary follows from Theorem 2.

*Assumption C.2**. The distribution $P(\boldsymbol{x}, y)$ satisfies that $P(Y = +1|\boldsymbol{X} = \boldsymbol{x}) = \frac{w(-)\ell'(-\boldsymbol{x}^T\boldsymbol{\beta}^*)}{w(-)\ell'(-\boldsymbol{x}^T\boldsymbol{\beta}^*)+w(+)\ell'(\boldsymbol{x}^T\boldsymbol{\beta}^*)} = p_0(\boldsymbol{x}^T\boldsymbol{\beta}^*)$, where $\boldsymbol{\beta}^*$ is the global minimizer of $E[W_Y\ell(Y\boldsymbol{X}^T\boldsymbol{\beta})]$ that does not depend on $\boldsymbol{X}$. Here $W_Y = w(+)$ if $Y = 1$ and $w(-)$ otherwise.

*Assumption C.3**. The estimated $\hat{\boldsymbol{\beta}} = \text{argmin}_{\boldsymbol{\beta}}[\frac{1}{n}\sum_{i=1}^n W(y_i)\ell(y_i\boldsymbol{x}_i^T\boldsymbol{\beta}) + \lambda J(\boldsymbol{\beta})]$ satisfies $\hat{\boldsymbol{\beta}} \to \theta\boldsymbol{\beta}^*$ in probability as $n \to \infty$, where $\theta \in (0, 1)$.

Corollary 1. For linear learning, suppose that Assumptions A.1, C.1, C.2*, and C.3* are satisfied. Then the estimates in Step 3 satisfy $\hat{\gamma}_0 \to 0$ and $\hat{\gamma}_1 \to \frac{1}{\theta}$ in probability, as $n \to \infty$.

4. SIMULATION

In this section, we use two simulated examples to illustrate the performance of PLR and PSVM. We compare the original and the refit methods for probability estimation. In both examples, we set the dimensions of covariates to be 5, 50, 100, 250, and 500.

Table 1. The average classification and probability estimation errors using PLR for Example 1 with $n = 100$. The corresponding standard errors are reported in parentheses. The LR cannot be calculated for $d \geq 50$ due to numerical difficulties, and we use NA for those entries

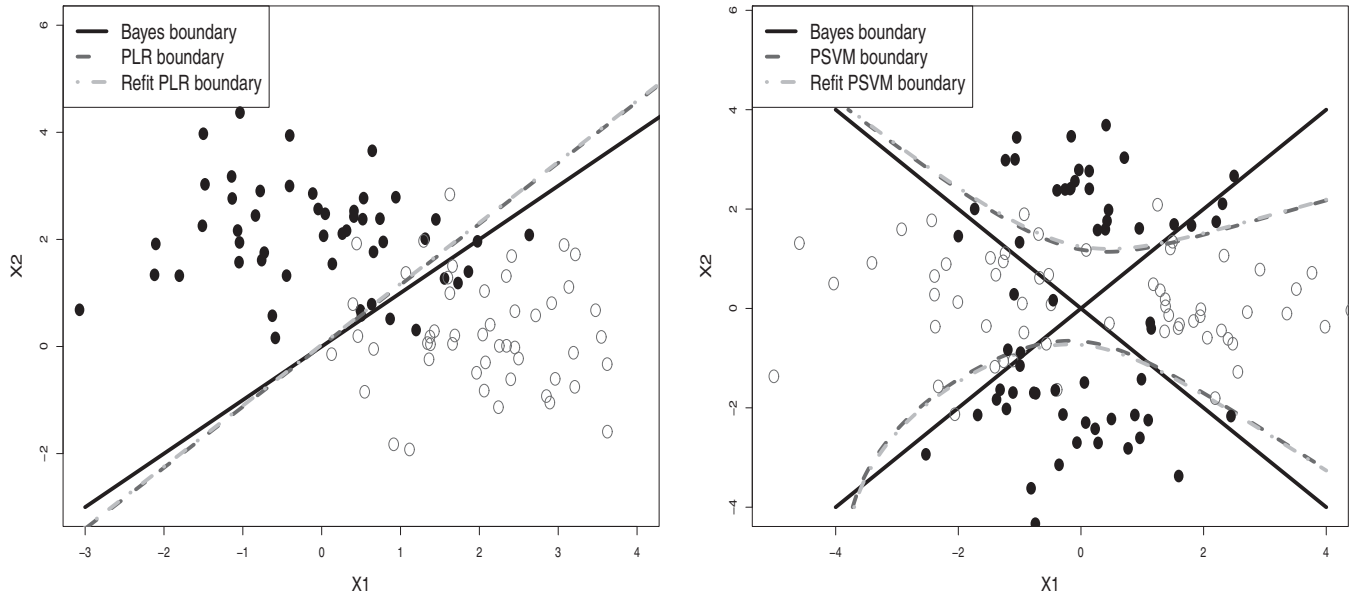
	Dimension	5	50	100	250	500
mean($\lvert p^{\text{true}} - \hat{p}\rvert$)	Original PLR	0.2302 (0.00077)	0.2698 (0.00057)	0.3112 (0.00023)	0.4092 (0.00019)	0.4236 (0.00076)
	Refit PLR	0.0452 (0.00132)	0.1026 (0.00188)	0.1479 (0.00255)	0.1717 (0.00198)	0.2064 (0.00219)
	LR	0.0912 (0.00257)	NA	NA	NA	NA
Misclassification error	Original PLR	0.0847 (0.00067)	0.1259 (0.00126)	0.1621 (0.00190)	0.2321 (0.00200)	0.2874 (0.00171)
	Refit PLR	0.0851 (0.00059)	0.1258 (0.00152)	0.1603 (0.00237)	0.2325 (0.00290)	0.2895 (0.00177)
	LR	0.1089 (0.00223)	NA	NA	NA	NA

Example 1. In this example, we generate the data as follows. For the positive class, the 1st and 2nd covariates follow $N[(2, 0)^T, I_2]$. For the negative class, the corresponding distribution is $N[(0, 2)^T, I_2]$. The remaining $d - 2$ covariates follow iid$N(0, 1)$, where d is the dimensionality of $\boldsymbol{x}$.

The training data have 100 observations. The tuning parameter λ is chosen using a grid search. Specifically, for each candidate λ value in $\{2^{-10}, 2^{-9}, \ldots, 2^{39}, 2^{40}\}$, we fit the model with PLR, and obtain the misclassification error rate on a separate tuning dataset. The tuning dataset has 100 observations and it is generated in the same way as the training dataset. We choose the λ value that corresponds to the minimal error rate. A different test set of size 10^4 is used to evaluate the performance of both classification accuracy and probability estimation. We repeat the whole procedure 1000 times to calculate the average misclassification rate, $P(Y \neq \hat{Y})$, and probability estimation error, $\frac{1}{\#(\text{test set})} \sum_{i \in \text{ test set}} (|\hat{p}_i - p_i^{\text{true}}|)$. We report average probability estimation errors and average misclassification rates in Table 1. The corresponding standard errors are reported in parentheses. As a comparison, the regular logistic regression is also used here.

We can see from Table 1 that the absolute difference between p_{true} and $\hat{p}$ for the refit method is much smaller than the original estimator. This demonstrates the effectiveness of our refit method. In terms of classification errors, the refit method is almost identical to the original fit. As an illustration, we plot the classification boundaries before and after the refit step, along with the Bayes' boundary, on the left panel of Figure 1, for one typical simulation. We can see that the refit step does not change the boundary much.

Example 2. This is a nonlinear example. Similar to Example 1, only the first two covariates are relevant for classification.



(a) Example 1, classification boundaries

(b) Example 2, classification boundaries

Figure 1. The left panel shows the classification boundaries of the original and the refit methods for PLR in Example 1. The right panel shows the classification boundaries of the original and the refit methods for PSVM in Example 2. Clearly the classification boundaries of the original and the refit methods are almost identical. (a) Example 1, classification boundaries. (b) Example 2, classification boundaries.

Table 2. The average classification and probability estimation errors using PSVM for Example 2 with $n = 120$. The corresponding standard errors are reported in parentheses

	Dimension	5	50	100	250	500
mean($\lvert p^{\text{true}} - \hat{p}\rvert$)	Original PSVM	0.1892 (0.00030)	0.2011 (0.00019)	0.2473 (0.00024)	0.3019 (0.00054)	0.3698 (0.00033)
	Refit PSVM	0.0801 (0.00377)	0.1195 (0.00560)	0.1278 (0.00573)	0.1503 (0.00399)	0.1610 (0.00298)
Misclassification error	Original PSVM	0.1621 (0.00199)	0.1939 (0.00356)	0.1944 (0.00511)	0.1957 (0.00299)	0.2016 (0.00318)
	Refit PSVM	0.1622 (0.00342)	0.1940 (0.00406)	0.1945 (0.00531)	0.1952 (0.00290)	0.2012 (0.00382)

For the positive class, the data points are generated as a mixture of normal distributions as $\frac{1}{2}N[(2,0)^T, I_2] + \frac{1}{2}N[(-2,0)^T, I_2]$, and the negative class is from a different mixture of normal distributions as $\frac{1}{2}N[(0,2)^T, I_2] + \frac{1}{2}N[(0,-2)^T, I_2]$. To achieve nonlinear learning, we add the second- and third-order polynomial terms of the 1st and 2nd covariates as additional covariates into the original data, and then apply linear learning using the PSVM. The training sample size is set to be 120. We generate 120 tuning observations in a similar manner as in Example 1. The tuning parameter λ is chosen in the same way as in the previous example, and the number of replications is also 1000. We report the probability estimation errors and the misclassification rates in Table 2. The Bayes' boundary, the classification boundaries before and after the refit step are reported on the right panel of Figure 1. A similar conclusion as in Example 1 can be drawn from the results in Table 2 and the right panel of Figure 1. The refit helps to improve probability estimation without sacrificing the classification accuracy.

5. REAL DATA

In this section, we investigate the performance of our proposed refit method on the dataset ionosphere. For this ionosphere data, the goal is to clarify good versus bad radar returns using 34 input attributes. Good radar returns are those showing evidence of certain structure in the ionosphere, and bad returns are those that do not. There are 351 samples in total. More information about this dataset can be found on the UCI machine learning repository database website, *http://www.ics.uci.edu/mlearn/MLRepository.html*.

Similar to that of the simulated examples, we randomly choose 70 observations for training, 75 for tuning, and the remaining for testing (see Hastie, Tibshirani, and Friedman 2009 for more discussion on model selection and model evaluation). In each random split, λ is chosen by a grid search as in the simulated examples, and we compare the probability estimation of the original methods with that of the refit methods. We apply PLR for both datasets. Since the underlying probability distribution is unknown, we evaluate the closeness of $\hat{p}$ to p in terms of the Cross Entropy Error (CRE; Wang, Shen, and Liu 2008), where $\text{CRE}(\hat{p}) = -\frac{1}{\#(\text{test set})}\sum_{\text{test set}} \{\frac{1}{2}(1+y_i)\log[\hat{p}(x_i)] + \frac{1}{2}(1-y_i)\log[1-\hat{p}(x_i)]\}$. Note that some other criteria such as Brier scores can be used as well. We standardize the input covariates before the analysis, and apply linear learning. The CREs of the original estimator and of the refit method based on 1000 times of random splitting are 3.211(0.062) and 0.525(0.006), respectively. This suggests that the refit method improves probability estimation significantly for this example.

6. DISCUSSION

In this article, we investigate the problem of probability estimation for large margin classifiers and illustrate the bias problem on class probability estimation created by shrinkage. We show that such bias can be large for finite sample problems. As a result, alternative procedures are needed. Our simple refit method helps to correct the scale problem introduced by shrinkage and yields accurate class probability estimation.

As a remark, we would like to mention that the work of Zhu and Hastie (2003) provides a promising path for further improvement of probability estimation. In particular, they proposed an interesting idea of feature selection through density estimation. For our case, suitable feature selection via density estimation may help to yield a more flexible and robust projection space for the refit step.

Our focus in this article is on binary classification. For multicategory problems, we believe similar phenomena exist and corrections are necessary as well. Since there will be multiple classification functions, the projection step is more complicated. Further investigation will be pursued.

APPENDIX: PROOFS

Proof of Theorem 1. Notice that $\min E\{\ell[Yf(\boldsymbol{X})]\} = \min E\{E[\ell[Yf(\boldsymbol{X})]|\boldsymbol{X} = \boldsymbol{x}]\}$. Letting $S = E\{\ell[Yf(\boldsymbol{X})]|\boldsymbol{X} = \boldsymbol{x}\} = p(\boldsymbol{x})\ell[f(\boldsymbol{x})] + [1-p(\boldsymbol{x})]\ell[-f(\boldsymbol{x})]$, we have $\frac{\partial S}{\partial f}|_{f=f^*} = \ell'(f^*)p - \ell'(-f^*)(1-p) = 0$. As $\ell'(u) + \ell'(-u) < 0$ for $\forall u$, $p(\boldsymbol{x}) = \frac{\ell'[-f^*(\boldsymbol{x})]}{\ell'[f^*(\boldsymbol{x})]+\ell'[-f^*(\boldsymbol{x})]}$. Now taking the derivative of $p(\boldsymbol{x})$ with respect to $f^*(\boldsymbol{x})$ yields $\frac{dp}{df^*} = -\frac{\ell'(f^*)\ell''(-f^*)+\ell'(-f^*)\ell''(f^*)}{[\ell'(f^*)+\ell'(-f^*)]^2}$. Thus by Condition B.3, $p(\boldsymbol{x})$ is a strictly increasing function of $f^*(\boldsymbol{x})$, which guarantees a one-to-one correspondence between them. □

Proof of Theorem 2. Let $\eta_i = \boldsymbol{x}_i^T\boldsymbol{\beta}^*$. Recall that $\hat{\eta}_i = \boldsymbol{x}_i^T\hat{\boldsymbol{\beta}}$. The empirical loss function we minimize in the refit step is

$$\min_{\gamma_0,\gamma_1} \frac{1}{n}\sum_{i=1}^{n} \ell[y_i(\hat{\eta}_i\gamma_1 + \gamma_0)],$$

and we define

$$L(\hat{\gamma}_0, \hat{\gamma}_1) := \frac{1}{n}\sum_{i=1}^{n} \ell[y_i(\hat{\eta}_i\hat{\gamma}_1 + \hat{\gamma}_0)].$$

Assume that $\hat{\gamma}_0$ does not converge to 0 in probability, as $n \to \infty$. We then have a subsequence of $\hat{\gamma}_0$'s that converges to another real number $z \neq 0$ in probability. For simplicity, assume that the entire sequence $\hat{\gamma}_0$ converges to z. Note that as $n \to \infty$, $L(0, \frac{1}{\theta})$ converges to $E[\ell(y\eta)]$ by Assumption C.3. Because $L(\hat{\gamma}_0, \hat{\gamma}_1)$ does not converge to $E[\ell(y\eta)]$ for any choice of $\hat{\gamma}_1$ if $\hat{\gamma}_0 \to z$, we can conclude that for large enough n, $L(\hat{\gamma}_0, \hat{\gamma}_1) > L(0, \frac{1}{\theta})$, by Assumption C.2. This contradicts the fact that $(\hat{\gamma}_0, \hat{\gamma}_1)$ is the minimizer of L. Thus, $\hat{\gamma}_0$ converges to 0 in probability, as $n \to \infty$. A similar argument can show that $\hat{\gamma}_1$ converges to $\frac{1}{\theta}$ in probability, as $n \to \infty$. This completes the proof. □

Proof of Corollary 1. In weighted learning, the empirical loss function we minimize in the refit step is

$$\min_{\gamma_0,\gamma_1} \frac{1}{n}\sum_{i=1}^{n} W(y_i)\ell[y_i(\hat{\eta}_i\gamma_1 + \gamma_0)],$$

and now the definition of L becomes

$$L(\hat{\gamma}_0, \hat{\gamma}_1) := \frac{1}{n}\sum_{i=1}^{n} W(y_i)\ell[y_i(\hat{\eta}_i\hat{\gamma}_1 + \hat{\gamma}_0)].$$

The rest of the proof follows the same line as that of Theorem 2, except that $L(0, \frac{1}{\theta})$ converges to $E[W(y)\ell(y\eta)]$ instead of $E[\ell(y\eta)]$. □

[Received May 2012. Revised June 2013.]

REFERENCES

Bartlettf, P. L., Jordan, M. I., and McAuliffe, J. D. (2006), "Convexity, Classification, and Risk Bounds," *Journal of the American Statistical Association*, 101, 138–156. [136]

Cristianini, N., and Shawe-Taylor, J. (2000), *An Introduction to Support Vector Machines*, Cambridge: Cambridge University Press. [135,137]

Freund, Y., and Schapire, R. E. (1997), "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, 55, 119–139. [134]

Friedman, J., Hastie, T., and Tibshirani, R. (2000), "Additive Logistic Regression: A Statistical View of Boosting," *The Annals of Statistics*, 28, 337–407. [135,136]

Fung, G., and Mangasarian, O. L. (2001), "Proximal Support Vector Machine Classifiers," in *Proceedings KDD-2001: Knowledge Discovery and Data Mining*, pp. 77–86. [135,136]

Hastie, T. J., Tibshirani, R. J., and Friedman, J. (2009), *The Elements of Statistical Learning*, New York: Springer. [134,135,136,141]

Lin, X., Wahba, G., Xiang, D., Gao, F., Klein, R., and Klein, B. (2000), "Smoothing Spline ANOVA Models for Large Data Sets With Bernoulli Observations and the Randomized GACV," *The Annals of Statistics*, 28, 1570–1600. [135]

Lin, Y. (2000), "Tensor Product Spave Anova Models," *The Annals of Statistics*, 28, 734–755. [135,136]

——— (2002), "Support Vector Machines and the Bayes Rule in Classification," *Data Mining and Knowledge Discovery*, 6, 259–275. [134]

——— (2004), "A Note on Margin-Based Loss Functions in Classification," *Statistics and Probability Letters*, 68, 73–82. [136]

Lin, Y., Lee, Y., and Wahba, G. (2002), "Support Vector Machine for Classification in Nonstandard Situation," *Machine Learning*, 46, 191–202. [137]

Liu, Y., Zhang, H. H., and Wu, Y. (2011), "Soft or Hard Classification? Large Margin Unified Machines," *Journal of the American Statistical Association*, 106, 166–177. [135]

Meinshausen, N. (2007), "Relaxed Lasso," *Computational Statistics and Data Analysis*, 52, 374–393. [138]

Qiao, X., and Liu, Y. (2009), "Adaptive Weighted Learning for Unbalanced Multicategory Classification," *Biometrics*, 65, 159–168. [138]

Qiao, X., Zhang, H. H., Liu, Y., Todd, M. J., and Marron, J. S. (2010), "Weighted Distance Weighted Discrimination and its Asymptotic Properties," *Journal of the American Statistical Association*, 105, 401–414. [138]

Schölkopf, B., and Smola, A. J. (2002), *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*, Cambridge, MA: The MIT Press. [137]

Shen, X., Tseng, G., Zhang, X., and Wong, W. (2003), "On ψ-Learning," *Journal of the American Statistical Association*, 98, 724–734. [135]

Suykens, J. A. K., and Vandewalle, J. (1999), "Least Squares Support Vector Machine Classifiers," *Neural Processing Letters*, 9, 293–300. [135]

Tang, Y., and Zhang, H. H. (2005), "Multiclass Proximal Support Vector Machines," *Journal of Computational and Graphical Statistics*, 15, 339–355. [135]

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society,* Series B, 58, 267–288. [138]

Wahba, G. (1999), "Support Vector Machines, Reproducing Kernel Hilbert Spaces and the Randomized GACV," in *Advances in Kernel Methods Support Vector Learning*, eds. B. Schölkopf, C. J. C. Burges, A. J. Smola, MIT Press, pp. 69–88. [134,137]

Wang, J., Shen, X., and Liu, Y. (2008), "Probability Estimation for Large Margin Classifiers," *Biometrika*, 95, 149–167. [138,141]

Williams, E. J. (1959), *Regression Analysis*, New York: Wiley. [136]

Zhu, J., and Hastie, T. (2005), "Kernel Logistic Regression and the Import Vector Machine," *Journal of Computational and Graphical Statistics*, 14, 185–205. [135]

——— (2003), "Feature Extraction for Non-Parametric Discriminant Analysis," *Journal of Computational and Graphical Statistics*, 12, 101–120. [141]

Zou, H., Zhu, J., and Hastie, T. (2008), "New Multicategory Boosting Algorithms Based on Multicategory Fisher-Consistent Losses," *The Annals of Applied Statistics*, 2, 1290–1306. [136]